



JAN 15 1988
LIBRARIES

WORKING PAPER
ALFRED P. SLOAN SCHOOL OF MANAGEMENT

A FRAMEWORK AND COMPARATIVE STUDY
OF DISTRIBUTED HETEROGENEOUS
DATABASE MANAGEMENT SYSTEMS

Subhash Bhalla
Bandreddi E. Prasad
Amar Gupta
Stuart E. Madnick

February 1988

#WP 1981-88

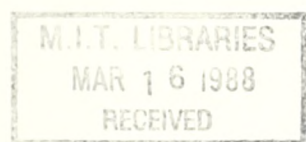
MASSACHUSETTS
INSTITUTE OF TECHNOLOGY
50 MEMORIAL DRIVE
CAMBRIDGE, MASSACHUSETTS 02139

A FRAMEWORK AND COMPARATIVE STUDY
OF DISTRIBUTED HETEROGENEOUS
DATABASE MANAGEMENT SYSTEMS

Subhash Bhalla
Bandreddi E. Prasad
Amar Gupta
Stuart E. Madnick

February 1988

#WP 1981-88



A FRAMEWORK AND COMPARATIVE STUDY OF
DISTRIBUTED HETEROGENEOUS DATABASE MANAGEMENT SYSTEMS

Subhash Bhalla*, Bandreddi E. Prasad**, Amar Gupta,
Stuart E. Madnick

Center for Information Systems Research
Sloan School of Management, Massachusetts
Institute of Technology, Cambridge, Mass. 02139

ABSTRACT

The prime objective of the Distributed Heterogeneous Database Management System approach is to support database integration across organizational, application, and geographical boundaries. This is achieved by efforts that at providing a unified global schema and common query facilities to users, without changing existing Database Management Systems or their application programs.

Design methodologies for such systems differ from each other in a number of ways. The additional complexity of translating between multiple systems and data models makes Distributed Heterogeneous Database Management Systems more challenging than conventional database systems.

This paper identifies critical aspects of Distributed Heterogeneous Database Management Systems. It aims at providing a basis for the study of these systems, comparative analysis between such systems, and directions for further extensions.

*On leave from School of Computer and System Sciences,
Jawaharlal Nehru University, New Delhi-110067, India.

**School of Computer and Information Sciences, University of
Hyderabad, PO Central University, Hyderabad, India.

A FRAMEWORK AND COMPARATIVE STUDY OF DISTRIBUTED HETEROGENEOUS DATABASE MANAGEMENT SYSTEMS

Subhash Bhalla, Bandreddi E. Prasad, Amar Gupta,
Stuart E. Madnick

I. INTRODUCTION

By virtue of their size and their growing reliance on computerized data, large organizations must necessarily rely on multiple computer systems to support their operations. The need for multiple systems is dictated by:

- (a) the level of required processing power and the amount of storage space;
- (b) the desired level of reliability and fault-tolerance;
- (c) the geographic distribution of data collection, data manipulation, and data retrieval sites;
- (d) the decentralized and functional structure of the organization;
- (e) the diversity of operating needs of the organization; and
- (f) the need to deal with different types of information, each of which may favor use of particular classes of computer hardware and software facilities.

In virtually all large organizations, there are a number of dissimilar and incompatible hardware and software systems in operation. While these systems may meet the objectives for which each of them was initially designed, their heterogeneity presents a major obstacle in situations requiring access and assimilation

of information resident on dissimilar computing equipment.

In the current environment where many organizations are connecting their computational resources together using Local Area Networks and other communication links, there is a growing requirement for integrating existing data that reside on multiple systems (see figure 1). In order to meet this requirement, Distributed Database Management (DDBM) strategies have been explored.

DDBM systems are broadly classified into two categories: homogeneous systems and heterogeneous systems. In homogeneous systems, there is only one data model and one data manipulation language, typically from a single DBMS vendor. Unfortunately, these systems cannot meet the objectives of most organizations who already have many types of computers with different data models and multiple data manipulation languages. To meet such objectives and to minimize the impact of existing heterogeneity, it is necessary to use Distributed Heterogeneous Database Management Systems (DHDBMS). The overall purpose of a DHDBMS is to access, to aggregate and to update information maintained in existing, distributed, heterogeneous DBMSs through a single uniform interface without changing existing database systems and without disturbing local operations. Such a system performs the functions listed in Figure 2.

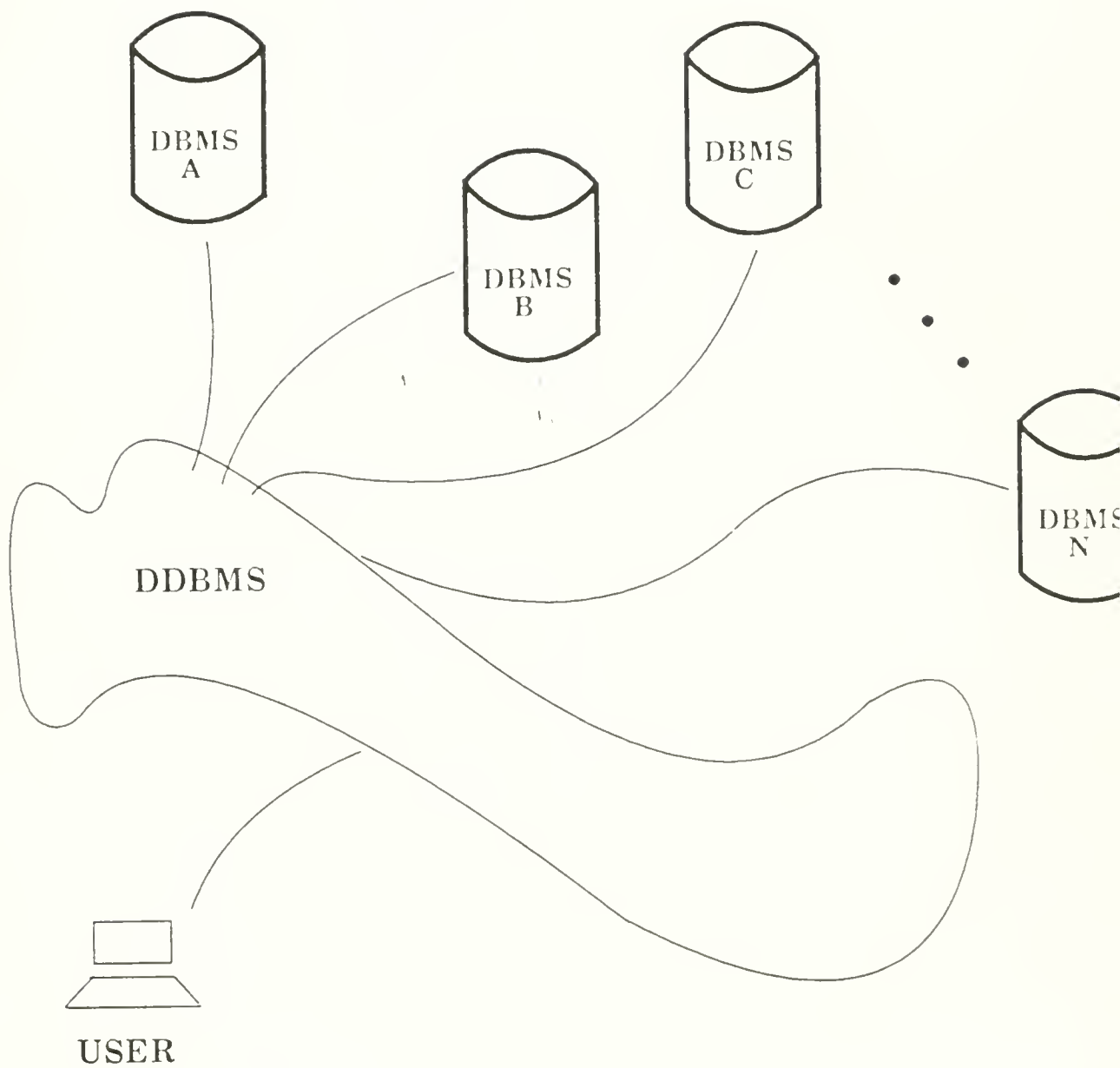


Figure 1 : Distributed Database Management System (DDBMS)

- (a) Accepts query and update requests made by user applications;
- (b) Transforms the requests into a set of subqueries expressed in the different languages supported by various local database management systems;
- (c) Formulates a plan for executing the sequence of subqueries and data movement operations;
- (d) Implements a plan for accessing data at individual local sites;
- (e) Resolves incompatibilities between the databases, such as differences in data types and conflicting schema names;
- (f) Resolves inconsistencies in copies of the same information that are stored in different databases; and
- (g) Combines pieces of retrieved data into a single response to the original request.

Figure 2. Functions of a Distributed Heterogeneous Database Management System (DHDBMS)

In order to perform these functions, within the constraints imposed by the existing set of systems, the major requirements are as follows:

- (a) Development of a Standard User Language and a Common Data Model;
- (b) Provision of facilities for Query Processing;
- (c) Incorporation of Distributed Transaction Management Routines;
- (d) Support of Distributed Operating System Functions and Network Services; and
- (e) Development of Authorization Control and Data Security procedures for the new decentralized environment.

These issues are examined in Section II of this paper. In Section III, various approaches are described using a set of eight representative prototypes. The salient features of these eight systems are summarized in Section IV. In the final section, recommendations for further research are formulated.

II. CRITICAL ASPECTS OF A DISTRIBUTED HETEROGENEOUS DBMS

The design of a DHDBMS is constrained by the following requirements:

- (a) The component systems are existing systems that were not originally designed to be part of a DDBMS;
- (b) The component systems cannot be easily modified; and
- (c) The component systems are not fixed permanently at design time. As such, it must be possible to freely add and remove systems to and from the DHDBMS.

The above requirements make it necessary to resolve conflicts at several levels. Some of these are :

- (a) Resolution of data structure conflicts caused by dissimilar data models used by different systems;
- (b) Resolution of naming conflicts, such as:
 - (i) semantically equivalent data items named differently in participating databases; and
 - (ii) semantically different data items having the same name in participating databases;
- (c) Resolution of data representation conflicts for the same data item in different databases. For example, a value may be represented as a character data type in one database, while it may be defined as a real data type item in another database;
- (d) Resolution of data scaling conflicts that arise due to

same data item being represented using different units of measure; and

- (e) Resolution of data inconsistencies for a data item residing in several databases.

The above goals are achieved by adopting a multiple schema architecture that provides heterogeneity transparency along with distribution transparency. The use of multiple schemas and the mappings between them serves as the mechanism for providing transparency across dissimilar systems and architectures [DEV82] [GLI84].

A global data model is used to capture the complete meaning of information stored at various Distributed Databases. All the data in the environment are defined in the global conceptual schema based on the global data model. This conceptual schema is mapped to many local file and DBMS structures (referred to as local schemata), and many user views (referred to as external schemata). Most prototype DHDBMS adhere to this three-schema approach to data integration [APP85].

We now turn our attention to query processing and query optimization. Both these processes are complicated because of the following factors [BRE84]:

- (a) Large size of the aggregate database;
- (b) Different local processing capabilities at the level of participating database systems;

- (c) Different communication costs of accessing local databases versus remote databases; and
- (d) Variable speeds of communication links.

The consistency of the aggregate set of databases is ensured through incorporation of Distributed Transaction Management Routines. Since existing systems contain dissimilar concurrency control mechanisms for handling local transactions, design of such routines is a complex process. Design and implementation of a global concurrency control mechanism requires synchronization of local transactions with non-local transactions that form part of global transactions. Unfortunately, most local component systems do not provide adequate capability to support such global operations.

Further, database management systems have traditionally been built on top of existing operating systems that were not designed with the requirements of DHDBMS in mind. A new DHDBMS has, therefore, to depend on local host operating systems for services such as buffering, file system management, and interprocess communication.

Data security acquires increased significance in the case of a DHDBMS. DBMS designers often assume that the identity of a user making a query is known, and that the protection scheme can be based on this identity. While this assumption is true for a centralized system, it is usually false in the case of a

distributed system.

II.1 DHDBMS ARCHITECTURE

In this section a general DHDBMS architecture is presented to provide a framework for discussion. Such a framework is useful for describing and comparing the features of existing DHDBMS prototypes, though individual prototypes may not match this architectural framework exactly.

Figure 3 shows the architecture of a DHDBMS. Each of the participating local hosts support a local DBMS (LDBMS). Each LDBMS provides facilities for creation of local schemas, based on its data model, called the local data model (LDM). The LDM is usually a traditional hierarchical, network or relational data model. The DHDBMS integration software needs to capture the entire data that exists at the local host systems, in order to provide integrated access. In order to achieve this, a global conceptual schema (also called a conceptual schema) is created based on a global data model (GDM). The global conceptual schema needs to capture the meaning of the total data in the environment in terms of objects, events and time-states, in addition to representing integrity constraints, relationships and dependencies. For such a reason, the data model adopted for conceptual schema design is usually based on the Entity-Relationship [CHE76] or semantic data model [SU 83] [DAY86]. A

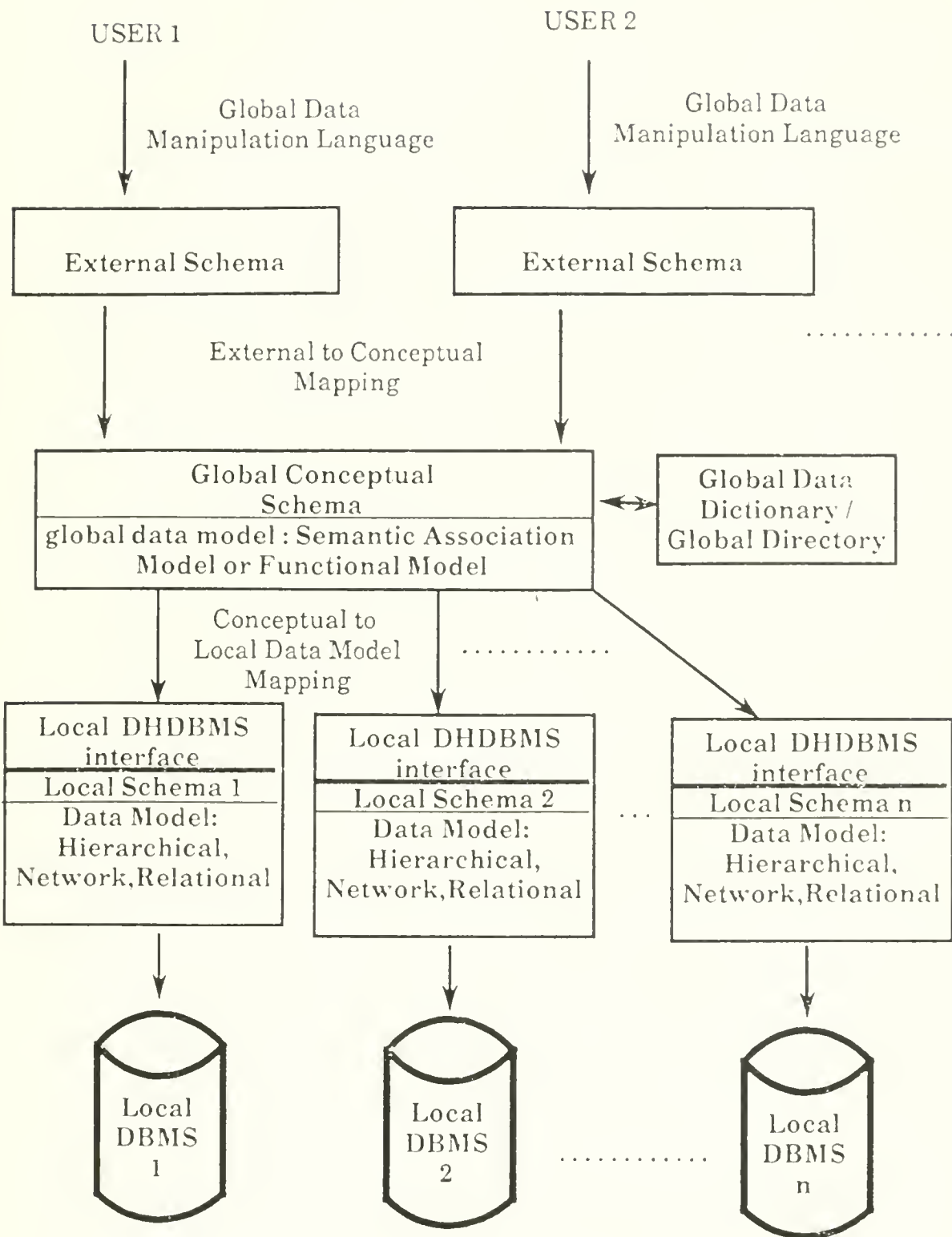


Figure 3: DHDBMS Architecture.

global conceptual schema also has an associated global data dictionary that describes the individual data components within the DHDBMS. In the case of a few DHDBMS prototypes (described below), the local database schemas are required to be redefined in order to support relational algebraic manipulation activity.

At the level of GDM, there is an associated global data manipulation language (GDML), such as DAPLEX in the case of MULTIBASE (see Figure 4.). The users express their queries in GDML and are provided with external schemas, based on the global conceptual schema. The user request is transformed to an intermediate query based on the conceptual schema. At this point a detailed plan of query execution is formulated and individual local hosts are accessed in order to process the sub-queries. The user request goes through another transformation, that is conversion between the conceptual schema and the local schema.

Figure 5 describes the functional architecture of a DHDBMS. The Command Processor accepts user requests. It performs the principal task of external to conceptual mapping referred in figure 3. It refers to the conceptual schema and the data dictionary to provide the global data management services such as, language transformation and global data integration.

The DHDBMS supports transaction management services that include global transaction processing and query management. These are provided by sub-systems consisting of the Decomposer, Merger

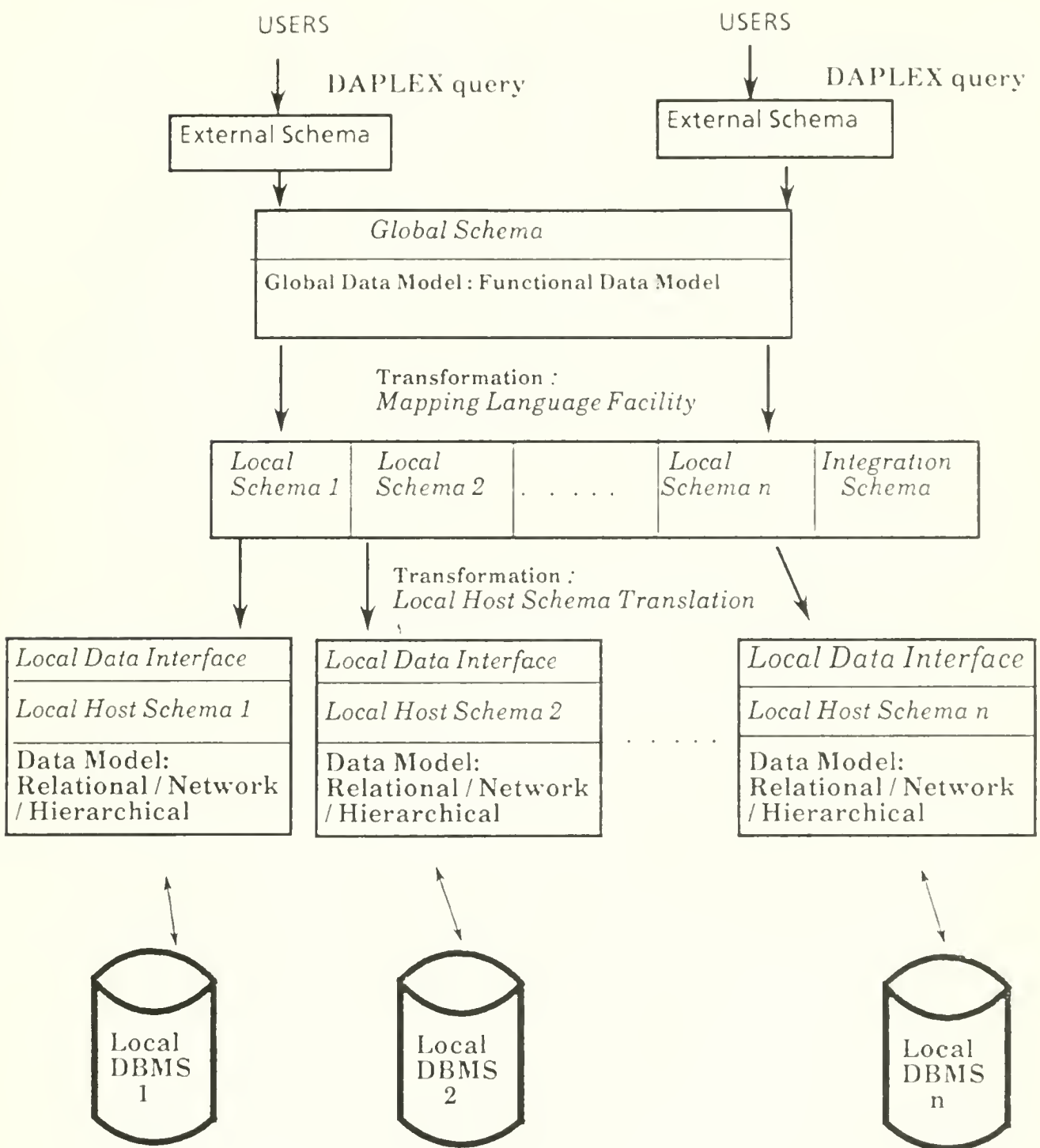


Figure 4: MULTIBASE Architecture

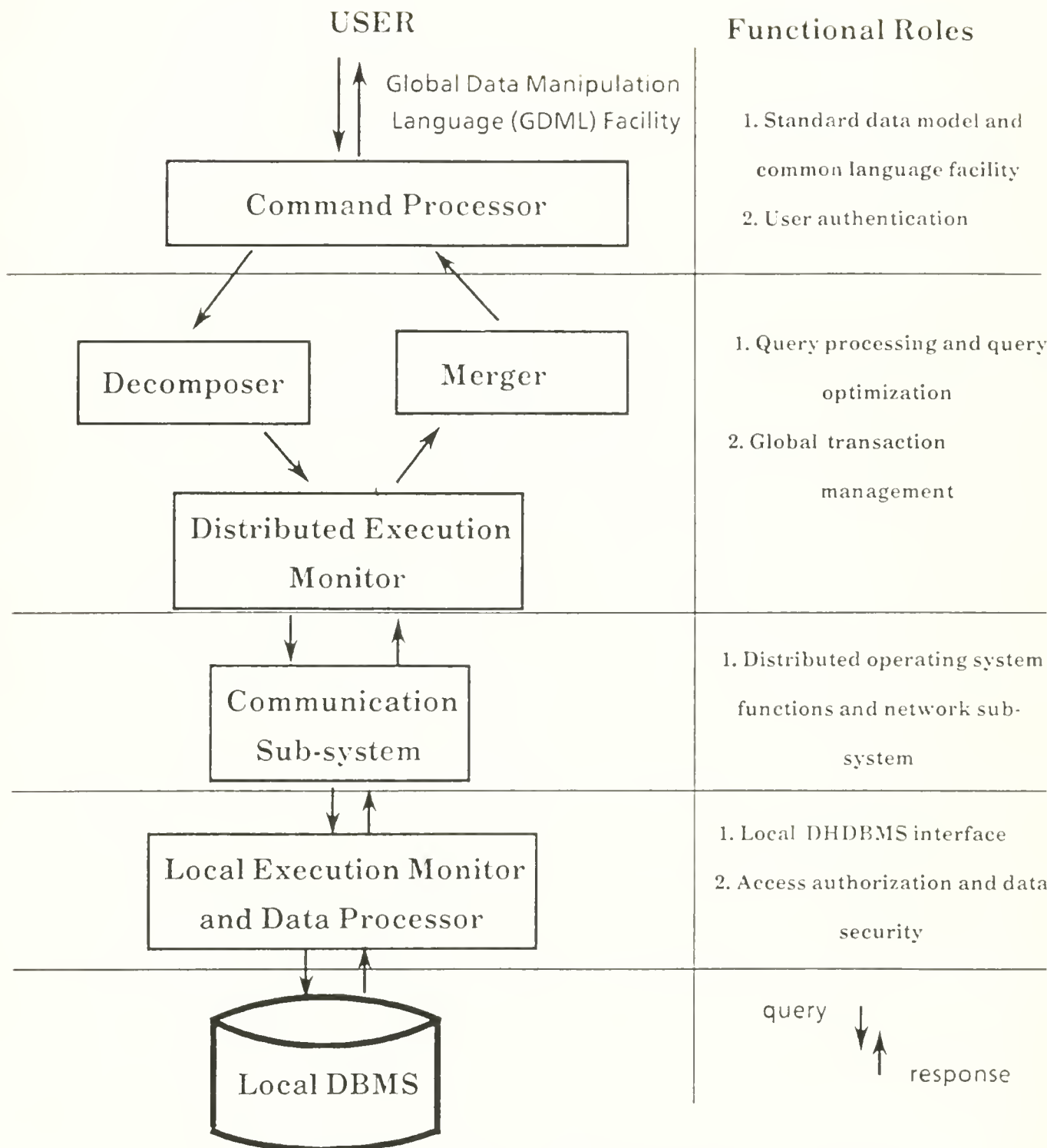


Figure 5: Functional architecture of distributed heterogeneous DBMS.

and Distributed Execution Monitor. The user request is interpreted by the Decomposer in terms of individual site access requirements. Similarly, Merger combines the results obtained as a result of query processing. The Distributed Execution Monitor performs the task of global transaction management. It access multiple DBMSs, depending on the access requirements of a query. It also performs query optimization. The individual components of data accessed at different sites, are refered to as base relations. In the case of update transactions, the desired goal of the global transaction management activity is to synchronize the updates, so that updating base relations leads to a new consistent state of the DHDBMS.

A Communication Sub-system provides the essential network communication services. The Local Execution Monitor and Data Processor performs the data transformation from the conceptual form to the local schema and supervises execution of the local processing steps. Also see figure 2 for associated roles connected with each level.

Most DHDBMSs are incorporated using existing computer communication servises, such as local area networks (LAN) or wide area networks. The DHDBMS interacts with the local DBMS through a local DHDBMS interface that resides at each local DBMS site.

The reference architecture described above is intended to serve as a model to show various levels and schemata that are

conceptually relevant. Individual DHDBMSs may have a single subsystem incorporating multiple levels indicated in the above model or may have multiple units incorporating a single reference level. In the next section, an overview of eight prototype systems is presented.

III. OVERVIEW OF SAMPLE PROTOTYPE SYSTEMS

DHDBMS prototypes differ significantly from each other in terms of their objectives, assumptions, and aims. Some of the prototypes are intended for specific applications, such as manufacturing, while others are general purpose; some assume large scale prevalence of relational databases in future systems, whereas others permit multiple types of databases; and some systems are designed to provide true distribution transparency, while others allow multiple global schemas, the main emphasis being on the autonomy of component databases. The different prototypes vary significantly in the techniques used to map local DBMS structures, to access data and to support intermediate models and languages. To understand and appreciate the diversity of goals and design approaches, eight prototype systems are studied in the following sections. The highlights of each system are described, followed by a comparison among these other systems.

MULTIBASE

MULTIBASE was developed by Computer Corporation of America, Cambridge, Mass., to provide a uniform, integrated interface for retrieving data from several existing DBMSs [LAN 82], [SMI81]. It allows a user to reference data in heterogeneous databases, through a common query language, using a single database description (Conceptual Schema) [DAY83] [GOL84].

MULTIBASE has been designed to serve as a general tool, without specific orientation towards any particular application area. It allows existing applications to operate without change and also permits new local systems to be included in an existing MULTIBASE system configuration.

The integrated access available through MULTIBASE does not provide either the capability to update the data in the local databases or the ability to synchronize read operations across several sites. In order to process user queries, the system must request and control specific services offered by the local systems (e.g., locking local items).

FUNCTIONAL ARCHITECTURE

MULTIBASE uses the language DAPLEX as its GDML. DAPLEX provides constructs that allow users to model real world situations in an efficient manner. A three level schema of definitions is employed, as previously shown in figure 4. The

process of global information retrieval involves two main components. These are : Global Data Manager (GDM), and Local Database Interface (LDI). The GDM performs tasks of Command Processor, Decomposer, Merger and Distributed Execution Monitor. The LDI in the case of MULTIBASE is designed based on the needs of a local DBMS, for example, a more sophisticated LDI is designed in order to support a file system as compared to the one that supports a DBMS. It provides the Local DHDBMS interface.

SALIENT FEATURES

To summarize, MULTIBASE provides an integrated scheme for:

- Uniform query access to dissimilar systems; and
- Global query optimization.

INTEGRATED MANUFACTURING DATA ADMINISTRATION SYSTEM (IMDAS)

The IMDAS architecture is being implemented as part of an experimental facility at the Automated Manufacturing Research Facility of the National Bureau of Standards. This testbed is intended to demonstrate the feasibility of supporting the manufacturing and production environment for factories of the future [BAR86], [LIB86]. The focus of IMDAS is on automating various functions related to manufacturing such as design, planning, and control. The main objective is to achieve a high level of software integration in an environment consisting of

engineering workstations, robots, and other machines, each operating on an autonomous basis. Supplementary objectives include :

- (a) Support for modular expansion, that is, support for network reconfiguration;
- (b) Effective resource utilization;
- (c) Efficient processing of time critical transactions, and replication of data to support such activities; and
- (d) Use of adaptive control techniques to intelligently react to failures and unexpected events.

FUNCTIONAL ARCHITECTURE

The global data model used by IMDAS, called SAM*, includes constructs for modeling the relationships among the data found in engineering, commercial, scientific and statistical databases. IMDAS supports three levels of schema definitions, as common to most heterogeneous distributed database management systems.

Between the two extremes of a centralized database management architecture and a distributed one, IMDAS has chosen a hybrid approach. IMDAS consists of three service layers, each of which is responsible for a definite set of distributed data management functions. These are Basic Data Administration System (BDAS), which includes local execution and local interface services; Distributed Data Administration System (DDAS) including

distributed query processing and transaction management services; and Master Data Administration (MDAS), that provides global data management services. These functions are distributed over the component systems according to their computational capabilities. The different layers of IMDAS software work together in establishing, manipulating, and controlling the distributed databases.

SALIENT FEATURES

IMDAS is being designed to achieve a high level of automation for factory environment. Currently work is in progress for implementation of global update processing.

INTEGRATED INFORMATION SUPPORT SYSTEM (IISS)

Integrated Information Support System (IISS), sponsored by the Wright-Patterson Air Force Base, attempts to support manufacturing and logistics operations of the U.S. Air Force [IIS83]. Its key design objectives are :

- (a) Efficiently utilize common data available in multiple DBMSs;
- (b) Support information resource management of various application systems in a closed-loop environment within manufacturing; and

(c) Provide access from geographically dispersed locations.

IISS uses LAN and wide area communication links to provide access to IBM 3081 (Network-IDMS), Honeywell level 6 (IDS/II), VAX (IDMS), and VAX (Relational - ORACLE).

FUNCTIONAL ARCHITECTURE

The three schema approach, comprising of External Schema, Global Conceptual Schema, and Internal Schema, are supported by a three schema data dictionary.

IISS permits two kind of processes : Integrated Application processes and Non-Integrated Application processes. In case of an Integrated Application process, a new application developed on IISS may access data which is distributed over several databases. A Non-Integrated application process may only access its local DBMS for purpose of retrieval or update. Further, a Network Trasaction Manager (NTM) has been implemented to provide sophisticated network services, such as interprocess communication through message passing.

SALIENT FEATURES

IISS plans to support global updates in future, in the case of Integrated Application processes. The current implementation suffers from the lack of interactive query processing support.

PROTOTYPE OF A RELATIONAL CANONICAL INTERFACE (PRECIS)

A prototype of a generalized distributed database system called PRECIS^{*} is being developed at the University of Keele and at the University of Aberdeen, in collaboration with a number of research centers, mainly in Britain [DEE87a] [DEE85]. It is a generalized DHDBMS, with both retrieval and limited update facilities. Existing databases are referred to as nodes upon redefinition of their local schema for access via a relational algebraic interface.

FUNCTIONAL ARCHITECTURE

PRECIS^{*} is based on a canonical data model. It also uses extended ANSI/SPARC architecture, and its conceptual schema (called canonical schema) has been written in a relational form. The principal data manipulation language is PRECIS Algebraic Language (PAL), which offers commands, such as Alteration, Rename, and Change Scale, for data integration [DEE87b].

Each nodal database in PRECIS^{*} is fully autonomous, with its own nodal DBMS (NDMS) and nodal external schema (NES). The latter provides a PAL interface to the distributed database which uses PAL as the standard language for communications. The different PRECIS^{*} schema levels include:

- External schema called the global external schema (GES)

which supports user views;

- Global conceptual schema called the global database schema (GDS) which is formed by the collection of participation schemas (PSs);
- Local DHDBMS interfaces known as participation schemas (PS) which describe nodal data with authorization controls. These are associated with either inner nodal database schema (NDS) or outer nodal database schema (NDS), as explained below.
- Local schema called the nodal external schema (NES).

SALIENT FEATURES

PRECIS* allows participation of a large number of local DBMSs as nodes. A node may participate in a network either as an inner node or as an outer node. The inner nodes contribute to the Global Conceptual Schema definition. If the number of participating nodes is large, some of these nodes are designated as outer nodes. These nodes do not contribute to the conceptual schema (GDS). The users at these nodes are permitted to have partially integrated view by defining their own mappings. Queries from users at the outer nodes are dealt with in much the same way as from others at the inner nodes. This strategy reduces the overhead involved in creating GDS and GES for a large number of nodes.

The Local Database Schema must be redefined to support relational algebra or PAL. PRECI[®] allows global updates on base relations only. That is, global update requirements are submitted to local DBMSs on individual database basis. If the data is replicated, update is performed only on the original copy and the result is broadcasted to other copies.

A DISTRIBUTED DATABASE SYSTEM (ADDS)

ADDS, developed by Amoco Production Company, provides a uniform interface to existing heterogeneous DBMSs located at various nodes of a computer network [BRE84]. For specific oil exploration and production projects, data is extracted from the IMS databases, sent to the project location, merged with the local data, and stored locally in relational and pseudo-relational databases, as a part of regular data extraction and data merge operations. The user is provided with a logically integrated view of the database and queries can be formulated using relational algebra operations over a predefined set of relations.

FUNCTIONAL ARCHITECTURE

The ADDS system includes:

- (a) Local DBMSs called the Physical Databases (PDBs) which are databases that exist on a computer network node;

- (b) Local DHDBMS interfaces called the Logical Databases (LDBs); and
- (c) Global conceptual schema called Composite Databases (CDBs) which contain a collective view for a set of LDBs, that constitute a single database from the designer's point of view.

Each CDB is associated with a directory, where ADDS schema definition, CDB information, and various user views of the CDB are recorded. This directory is maintained as a relational database.

The DHDBMS supports, a User Interface (UI) subsystem, which performs the functions of Command Processor, Decomposer and Merger. A subsystem called the Task Master (TM) within ADDS, performs the Distributed Execution Monitor function. The local DHDBMS interface is provided by a component called the Request Manager (RM).

SALIENT FEATURES

ADDS provides a relational data model based, integrated access interface to existing DBMSs. In addition to providing the user with a universal view (a relation of logical fields), the query language of ADDS also provides facilities for a relational view, which expresses the CDB as a set of PDBs and their logical fields. In this manner, it provides a range of query capabilities

for users of varying sophistication.

MULTICS RELATIONAL DATA STORE MULTIBASE (MRDSM)

MRDSM extends the MRDS relational database management system of HONEYWELL to support multiple databases. MRDSM is being developed by INRIA (France). It operates on a specialized domain of multiple MRDS relational DBMSs running on HONEYWELL systems [LIT85], [LIT86], [WON84]. It is not a true heterogeneous system because heterogeneity is only a concern at the semantic level, since all databases are implemented with the same DBMS. The query language is MDSL (similar to SQL), which is also the data manipulation language for MRDS.

FUNCTIONAL ARCHITECTURE

A Global Schema does not exist in MRDSM. Instead, users can create a conceptual schema known as a multischema with elements from local database schemas. Multischema is also associated with one or more Dependency Schemas which provide details such as inter-database dependencies. These include replication details as well as naming information.

A query on a multischema is decomposed into queries on local databases after removing inter-data dependencies that cannot be handled locally. A working Database Schema is then created to collect data from different databases. The collection process has

been optimized. Queries are then generated on the working databases. The final step is to combine all streams of data together and resolve dependencies.

MERMAID

MERMAID is an integrated data access system being developed by UNISYS [TEM87]. It allows users of multiple databases (relational DBMSs), running on different machines to manipulate data using a common language, which can be either ARIEL or SQL [MAC85], [TEM86], [YU 85].

The MERMAID integrated access system has been implemented using VAX (IDM, Britton-Lee), SUN 170 (INGRES), SUN 120(INGRES), and SUN 120(MISTRESS). Presently the system permits updates to a single database on an individual database basis, as in the case of PRECI*.

FUNCTIONAL ARCHITECTURE

The major processors of the MERMAID system are as follows:

- (a) The User Interface Processor: It contains an embedded ARIEL or SQL parser and a translator that produces DIL(Distributed Intermediate Language);
- (b) The Distributor Processor: It contains an optimizer and a controller;
- (c) DBMS Driver Processor, one for each database to be

accessed. This driver can also translate from DIL to the DBMS query language.

All information about schemata, databases, users, host computers, and the network is contained in a DD/D(Data dictionary/Directory) which is accessed through a special driver. It supports three layers of schema definitions.

The translator component within the User Interface Processor performs the functions of a Command Processor and a Decomposer. To process a query in ARIEL or SQL, the translator, parses and validates the query and passes it to the distributor. The Distributed Execution Monitor function is performed by the distributor. The controller part of the distributor reads the query in DIL and passes it to the optimizer part which plans the execution. The DIL query is decomposed into several subqueries and the controller sends them to one or more DBMS drivers for execution.

SALIENT FEATURES

MERMAID is an operational prototype which demonstrates the feasibility of operating as a front-end to distributed relational DBMSs. A schema design tool is being developed which will support the user in developing the global view of a database from an existing local schema.

NETWORK DATA MANAGEMENT SYSTEM (NDMS)

NDMS is a system being developed by CRAI (Consorzio per la Ricerca e Applicazioni di Informatica, Italy) for the National Transport Informatic System of Italy [STA84]. It currently supports IDMS, ADABAS and RODAN DBMSs on IBM systems, and INGRES on a VAX system.

FUNCTIONAL ARCHITECTURE

NDMS supports the relational data model as the GDM. The three abstraction levels are the NDMS Internal Schema, the Application Schema (as the conceptual schema) and the End-user Views (external schema). The NDMS Internal Schema comprises of base relations defined as aggregations over the local database schema. The base relation definitions require data mappings to be specified for each local database, depending on its DBMS data model.

The local DHDBMS interface comprises of the NDMS control software and the System Encyclopedia. The System Encyclopedia contains all information pertaining to the respective node, user definitions, database mapping definitions, transaction definition, and the complete NDMS Internal Schema Definition.

The Node Data administrator is responsible for NDMS applications at each node. It defines relational views, using the SEQUEL view definition mechanism, as a collection of data

abstractions (aggregations and generalizations) over the NDMS internal schema. The NDMS version of SEQUEL has been modified to handle generalized abstractions. Defined relational views are available to users for defining their specific data abstractions.

SALIENT FEATURES

NDMS supports queued transactions and on-line transactions. Queued transactions are processed as local or remote batch processes. No exchange of messages is permitted for such transactions. On-line transactions are considered to be distributed transactions. The NDMS Transaction Processor provides facilities to invoke transaction programs, to support the user interface, to exchange messages between application programs, and to synchronize transaction commit operations. A System Journal is available to support recovery.

IV. COMPARISON OF SYSTEMS

The eight systems, described in section III, have been selected on the basis of their uniqueness and the level of technical information available. These systems are now compared against each other.

The major features of all the eight prototype systems are summarized in the Table 1. The approaches adopted by the eight representative systems are compared in the succeeding paragraphs.

	MULTIBASE	IMDAS	IISS	PRECIS	MERMAID	MRDSM	ADDS	NDMS
ACRONYM DERIVED FROM	None	Integrated Manufacturing Database Administration System	Integrated Information Support System	Prototype of a Relational Canonical Interface	None	Multics Relational Data Store Multibase	Amoco Distributed Database System	Network Data Management System
ENVIRONMENT	General	Manufacturing	Manufacturing and Logistics	General	General	General	General (mostly scientific and engineering databases relating to data collected from Oil Wells)	Transportation
ORGANIZATION	Computer Corporation of America (US)	National Bureau of Standards (US)	Air Force (US)	University of Keele (UK)	UNISYS, Formally System Development Corporation (US)	INRIA (France)	Amoco Production Company, Research (USA)	CRAI (Italy)
LOCAL DATA MODELS SUPPORTED	Hierarchical, Relational, Network	Relational	Relational, Network	Any model via relational algebra interface	Relational or Relational Interface	Relational or Relational Interface	Relational, Hierarchical and Network	Relational and Network
LOCAL DMS SUPPORTED (Current Version)	CODASYL database, Hierarchical database	Relational - INGRES, RIM, and (DB2 planned)	Network - IDMS on IBM 3081, IDS/II on Honeywell level 6, IDMS on VAX, Relational - ORACLE on VAX	Under Development	IDM (Britton-Lee) on VAX, INGRES on SUN 170 and SUN 120, MISTRESS on SUN 120, M204 under development.	Multiple MRDS on Honeywell	Hierarchical - IMS, INQUIRE Relational - SOLIDS, RIM, FOCUS; Some sequential file formats	Network - IDMS DB/DC and ADABAS on IBM; Relational - RODAM on IBM, INGRESS on VAX
GLOBAL DATA MODEL	Functional Data Model	Semantic Association Model (SAM*)	IDEF1 (ER Based)	Canonical Data Model	Relational	Extended Relational	Relational	Relational
GLOBAL DATA MANIPULATION LANGUAGE	DAPLEX. Local host may not support all capabilities provided by DAPLEX.	SQL-like. Supports interactive access and programs through attachment to local interprocess communication	No interactive query language. Query statements embedded in COBOL and precompiled.	PAL (PRECIS Algebraic Language) which is relational algebra based	SQL or ARIEL (SDC query language)	MDSL SQL like, extended version of DSL (the data manipulation language of MRDS)	Extended Relational Algebra Language, plus a subset of the ANSI SQL language	Modified version of SQL
LOCAL DATABASE SCHEMA CONVERSION	Local database schema called Local Host Schema (LHS) in any model needs to be redefined completely in Functional Data Model to enforce uniformity. New schema is called Local Schema (LS). Local host schema remains intact.	Local database schema redefined/extended into relational model with system utilities. Mapping work which would leave local host schema intact is progressing.	Redefined using Neutral Data Definition Language (NDDL). (No information about the structure of NDDL is available. It has been mentioned that it is capable of supporting relational and network schemas, entities and relations, and mapping)	Local database schema must be redefined to support relational algebra or PAL	Needs conversion into relational form. (System does not provide help for conversion from local data model to relational form or from relational form to local data model). Can map user relational view to repeating groups in local database.	No conversion required. The system is designed to serve databases implemented with MRDS DBMS on Honeywell machines. The aim is to deal with semantic heterogeneity in order to provide uniform access to these databases	Needs conversion into relational form. The schemas of the local databases are described as Physical Databases (PDBs). PDBs are comprised of Physical Database Components (PDBCs). A PDBC for a relational database is a relation. A PDBC for a hierarchical database may represent a path from the root segment of the database to a leaf segment.	Needs conversion to relational form. Has two parts: a) Base relations, and b) Semantic information mapping. Base relations are objects constructed with underlying database schema. Semantic and information about these objects is represented with Mapping Definition Language.
DATA INCOMPATIBILITIES AND SEMANTIC MISMATCHES	Resolved through an Integration Database Schema (called IS) for all the Local Schemas. Extent of coverage is not sufficient, but it can take care of differences like kilometers and miles, age in years and in qualitative form, (young, old), etc	Performs conversions, maintains relationships between dependent relations and parent relations. Resolved by Global Schema Mapping. Provided by Global Schema.	Performs data format and unit conversion operations.	Local database schemas converted to the relational form are placed directly in Global Schema. Integration data also placed separately in Global Schema. Unlike MULTIBASE, users can refer to incompatible information (e.g., kilometers and miles) through Global External Schema separately rather than as one unit. Mapping is provided by Global Schema.	Deals with two types of data translation schemes: functional type and enumerated type. Functional translation deals with problems like unit conversions (e.g., kilometers and miles) and format conversions (e.g., date) etc. Enumerated deals with converting sets of values through a table lookup (e.g., codes and names)	Handles three types of interdatabase dependencies. They are a) manipulation dependencies, b) privacy dependencies, c) equivalence dependencies. Equivalence dependencies handle data incompatibilities and semantic mismatches	Supports different names and characteristics (e.g., data types and units) assigned to semantically equivalent fields located in different databases through the system data definition language. Data conversion is performed as the physical data is loaded into the logical temporary relations	Maintains information along with Global Schema in System Encyclopedia. Further details are not available

	MULTIBASE	IMDAS	HISS	PRECIS*	MERMAID	MRDSM	ADDS	NDMS
GLOBAL SCHEMA CONSTRUCTION	Uses the Functional Data Model to define the global schema and the associated query language DAPLEX as the global data manipulation language. The local DBMS and their Local Host Schema (LHS) are mapped into local schema (LS) which are expressed in terms of the Global Data Model. The Local Schema are merged into global schema with an auxiliary integration Schema (IS) which contains information needed for reconciling inconsistencies between the local schemata.	On line SAM* Dictionary utilities for defining Global Schema, Views, Site Capabilities, Fragmentation, and Partitioning	The structure of NDDL is unknown. It is not clear how various local schemas, redefined in NDDL, are merged.	Converted Local Schemas in relational form are placed in Global Schema with node in information integration data and meta data are maintained in relational form in the Global Schema. Merging of mappings and related relations are maintained in Global External Schemas (Views).	All the relations from different local schema are grouped through a schema translation mechanism. Schema translation deals with many changes, partitions, one-to-many and many-to-one mapping of relations (e.g., a global relation 'ship' can be mapped to a relation with name 'boat' in one database and a join of two relations in another database). Local database schemas subject to schema and data translation mechanisms are maintained in Global Schema in relational form.	Global Schema does not exist. Users can create conceptual schema known as multi schema with elements from local database schemas along with one or more dependency schemas to handle inter database dependencies. The elements of multischema created by the user are: (i) one or more dependency schema, or (ii) explicit enumeration of local databases to compose a multischema.	A single Global Schema does not exist. Instead, a Composite Data Base Schema (CDBS) definition is used to describe the subset of all the physical databases that may be appropriate for an application. Many CDBS may be defined, depending on the number of applications. A CDBS is comprised of one or more Logical Data Bases (LDBs) and one or more Physical Databases (PDBs). An LDB is the set of PDBs that are managed by a single DBMS at a single site. All the CDBS are placed in a directory. A user can choose a CDB, define a view and query on this view. Also, a user can define his own CDB schema.	Local database schemas converted in relational form are further analyzed for naming conflicts and type conflicts. Equivalent base relations are integrated together.
QUERY PROCESSING (Retrieval)	A Query expressed in DAPLEX on Global Schema is decomposed into Queries on DAPLEX Local Schemas. These are further translated into Local DML queries. Adopts "no two variables in a query range over the same entity type" strategy to generate queries on Local Schemas. Adopts data reduction and data movement techniques in Global Query optimization strategy. Merging and final processing takes place at Global Schema result node.	A Query on Global External View is modified and converted to queries on Global Schema. If Global Schema Manager (called DDAS) cannot process the query, it is passed to the Master Schema Manager (called MDAS) for further processing; otherwise, it is translated into queries on local schemas.	Currently, query statements in Neutral Data Manipulation Language (NDML) on Global Schema are embedded in COROL and precompiled. On precompilation, source code files are sent to their respective hosts for compilation.	This system follows the same strategy as MULTIBASE except in the final processing. The designers have aimed at a higher degree of parallelism. While in MULTIBASE, final processing is done at a result node, PRECIS facilitates parallel processing at multiple nodes.	A query in SQL or ARIEL on Global Schema is translated into queries in an intermediate language called DIL (Distributed Intermediate Language), and later translated from DIL to the local DBMS language. Currently translators are available for IDL, QUEL and SQL. Query Processing supports fragmented and replicated relations. Final post processing is done by one of the hosts.	A query on multischema is decomposed into queries on local DBs after removing inter data dependencies that cannot be handled locally. Then a working DB is created to collect data from different DBs. The collection process is optimised by performing projection and selection operations on source DBs. Finally queries are generated on the working DB to combine data.	Queries can be submitted against any number of Composite Data Bases (CDB). A query is optimized for minimal communication cost and a subquery schedule is produced. The steps of the subquery schedules are processed by server against a base relation translated into the language of the local DBMS.	An Application Schema is defined at each node over internal schema that represents a group of users at that node. Users at that node define their views on this Application Schema. Queries posed on Application Schema are translated to local database language through Internal Schema. Query processing strategy has three parts: the intermediate storage structure, query optimization and query execution.
GLOBAL UPDATE AND TRANSACTION MANAGEMENT	Not Supported	Contains a component called Transaction Manager, to handle integrity, concurrency control and recovery. Allows Global update on base relations only. Recovery limited to cancel/restart. More work is in progress.	Committed to supporting global updates.	Allows Global update on base relations. Only if data is replicated, update is done only on the original copy and broadcasted to other copies (Not fully implemented as yet)	Update to single database is permitted.	Limited Multidatabase Update facility exists.	Conducting studies on the multidatabase update problem. ADDS retrieval system is being modified to evaluate new concurrency control algorithm.	Updates are done through co-ordinated distributed processing applications. Transaction programs need to be implemented as application programs in the host DBMS. Co-ordinated by Transaction Processor

	MULTIBASE	IMDAS	IISS	PRECI*	MERMAID	MRDSM	ADDS	NDMS
NETWORK SERVICES	Global Data Manager (GDM) and Local Data Interface (LDI) incorporate network interface for establishing communication.	Token bus, Ethernet LAN. TCP/IP, HDLC have plan to use MAP. Compatible network, inter-process communication through shared memory. Data Transactions - Data encoded in ANSI 1 (ISO 8824/8825 standard)	LAN and wide area communication. A kernel known as Network Transaction Manager (NTM) has been implemented to provide sophisticated services on the network. Interprocess communication through message passing.	Designers propose to develop a standard message oriented protocol for communications between distributed databases.	Ethernet, TCP/IP. Plan to use MAP and DODIS protocols for interprocess communication.	The system uses inter facilities over X-25 TRANSPAC Net (French national net) using Dial-out interface.	SNA, Ethernet (TCP/IP), Bypass. Uses a logical network approach to provide a common interface to the physical networks.	X-25 protocol. Inter-process communication through message passing
CURRENT STATE LIMITATIONS AND FUTURE DIRECTIONS	Internal prototype ready. U.S. Air Force has showed an interest in using it on all experimental basis in their applications. Currently permits READ-ONLY Access and local updates.	Global Schema Manager (called DDAS) and local schema manager (called BDAS) exist on UNIX and VMS. Work is in progress on Master Schema Manager. Research is on for coordination and synchronization of distributed transaction management functions	Feasibility demonstrated using a scenario selected by 22 contractors. Further work on interactive query facility and global update is pending.	Basic design is complete. Currently implementing a pilot system. Initial plan is to have two nodes at Aberdeen on Honeywell, one node at Belfast on VAX, and a fourth in Dublin on another VAX.	Not a commercial product. In use within DOD systems developed by UNISYS. Operates only on top of multiple relational DBMS. Tested on a Navy Database that contains ships, positions, weapons installed on the ships, etc. The database is distributed to four sites. Plan to extend into the areas of object management, security, and integration with deductive inference engine	Investigating application of knowledge processing techniques in the area of query processing, to simplify query expressions.	Current version operates under VM/CMS and accesses data controlled by IMS, SOLDS, RIM, FOCUS, and INQUIRE databases. The current version is being enhanced. The system includes a remote user interface and programming language interface for use on workstations. Future directions include supporting update transactions, ADDS to ADDS communication, tools for automatic CD8 creation, and support for additional DBMSs such as DB2.	Current prototype interfaces with DBMSs on IBM and VAX. Future extensions include graphic query interface, distributed query interface, design tools and performance monitoring. Testing on Transportation Information System.
REMARKS	Further work suspended due to lack of funds.	Designed to achieve a high level of automation for factory environment.	Designed to provide a high level of support for automated Manufacturing and Logistic environment.	Designed to handle replication of databases.	Claims to offer one of the most complete query optimization algorithms that has been implemented and tested. Support replicated and fragmented relations.	Operates in a specialized domain. Not a 'true' heterogeneous system. Heterogeneity is dealt at semantic level by providing uniform access to all the databases implemented with same DBMS.	Has been designed to work in the specialized environment of databases relating to oil wells. Current version is being enhanced for corporate wide deployment	Designed for meeting the requirements of National Transport Information System of Italy
CONTACT PLACE (Person)	Computer Corporation of America, Four Cambridge Center, Cambridge, MA 02142 (D. Smith)	Integrated Systems Group, National Bureau of Standards, Gaithersburg, MD 20899 (M. Mitchell, E. Hartmeyer, D. Libes or Howard Bloom)	Material Laboratory, Air Force, Wright Aeronautical Labs, Air Force Systems Command, Wright-Patterson AFB, OH 45433	Dept. of Computer Science, University of Keele, Staffs. ST5 5BG, England (S. M. Deen)	UNISYS, 2525 Colorado Ave., Santa Monica, California 90406 (M. Templeton)	IRPIA, BP 106, 78153 Le Chesnay Cedex, FRANCE (W. Litwin)	Amoco Production Company, Research, P.O. Box 3385 Tulsa, Oklahoma 74102 (Glenn R. Thompson)	CRAI, Via Bernini 5, Localita' S Stefano, 87036 Rende (Cosenza), ITALY (W. Staniskis)
REFERENCES	[SM81, LAN82, GOL84, DAY83, SH81]	[LH86, BAR86, KH85, SU83]	[H583]	[DE84, DE85, DEE81, DEE87a, DEE87b]	[YUB8, TEM87, TEM86a, MAC85]	[WON84, LIT85, LIT86]	[BRE84, BRE86]	[STA84]

IV.1 UNIFORM INTEGRATED ACCESS IN DISTRIBUTED HETEROGENEOUS DBMS

The task of providing integrated access for a DHDBMS involves providing a standard user language and a standard data model to permit global administration of data resident on systems as varied as large mainframe based hierarchical-IMS at one end and personal computer based DBASE-II relational database on the other.

In order to provide a uniform integrated access within a DHDBMS environment, most prototypes use a concept of three functional layers, which provide [GLI84]:

- (a) Global Data Management Service; inclusive of
 - (i) Standard User Language and Data Model
 - (ii) Query Processing and Query Optimization
- (b) Distributed Transaction Management Service; and
- (c) Network Service.

The different techniques used to provide such diverse capabilities are discussed below.

VI.1.1 STANDARD USER LANGUAGE AND DATA MODEL

A. Local Data Models

The eight prototypes are geared to support different sets of local data models which are summarized below. MULTIBASE currently

supports DBMSs with hierarchical or network data models [GOL84], though it can support any data model or even non-database file systems through proper interfaces; IISS and NDMS support DBMSs with relational and network data models [IIS83] [STA84]; IMDAS supports only relational local data models [LIB86]; ADDS supports both relational and hierarchical local data models [BRE84]; and PRECI* supports any model via relational algebra interface [DEE85]. Finally the two remaining prototypes MERMAID and MRDSM also support relational data models and other data models through a relational interface.

B. Global Data Models

MULTIBASE supports a Functional Data Model (and not a relational or a semantic data model) and an associated data manipulation language called DAPLEX [SHI81]; IMDAS uses a Semantic Association Model (SAM*) and a SQL-like query language [SU 83]; IISS supports IDEF-Extended, which is Entity-Relationship based, and queries are embedded in COBOL and precompiled [IIS83]; and PRECI* supports a Canonical Data Model with PAL (PRECI Algebraic Language) which is relational-algebra based [DEE84]. Other prototypes such as MERMAID, ADDS and NDMS support Relational Global Data Models with MRDSM using an extended relational global data model. The associated query languages for these prototypes are either SQL like or relational

algebra based [TEM86], [LIT85], [BRE86], [STA84].

C. Transformations

All the above data models are supported by Data Definition Languages (DDLs) that are used to define the global conceptual schema in terms of objects and events and to specify integrity constraints on relationships and dependencies [APP85]. The conceptual to external (user view) transformation of schemas is achieved through a global DML, which is usually relational-algebra oriented. Transformation of conceptual schema to local schema, on the other hand, involves translation of both structure and form for all the heterogeneous databases included in the system. This transformation is generally performed by software at each node and data are moved through the network in the conceptual schema form. By using a relational algebraic language for both queries and mappings, query decomposition is easier, as compared to data integration based on non-relational models or manipulation languages [DEE87b]. Prototype systems such as PRECI*, MERMAID, MRDSM, ADDS and NDMS have adopted this approach. Similarly, MULTIBASE adopts a three schema architecture including DAPLEX global schema with DAPLEX local schema for each participant plus a DAPLEX auxiliary schema, and local host schema for each site.

D. Data Dictionary

All the three schemata discussed above and the transformations between them are managed via a three-schema data dictionary. The existing local databases use data dictionaries which were designed for individual database management systems only. Most researchers are building their own three-schema data dictionary, e.g., IISS has a Common Data Model (CDM) subsystem [IIS83]. The CDM subsystem consists of two software modules : (i) the CDM dictionary, which is a database that describes the conceptual schema and the network environment, and (ii) the CDM processor which is the software, that accesses the CDM dictionary and transforms user's data requests into transactions that can be processed by local DBMSs.

The ADDS directory is centralized and contains information about all the active CDBs within the system [BRE84]. The directory is maintained as a relational database and provides flexible tools for its maintenance. IMDAS system has a concept of Data Directory Server that maintains and performs a metadata lookup to provide information about data location, data structure, and delivery information. The MERMAID system maintains a Data Dictionary/ Directory (DD/D), which is centrally stored. The DD/D contains information about the databases, the users, the DBMSs, the host computers, and the network. It supports the following four layers of schema definitions:

- (a) Subschema layer: It represents the user's view based on the global schema;
- (b) Global schema: It contains the federated view of all the data definitions in the distributed global schema;
- (c) Distributed local schema: It represents the relational view of the local schema; and
- (d) Local schema: It corresponds to the external view of the local database.

Most of the prototypes maintain the data dictionary as a centralized resource at one site and all distributed database managers access this central data dictionary. For further information on local database schema conversion, data incompatibility and semantic mismatch handling, and global schema construction, please see Table 1.

IV.1.2 QUERY PROCESSING AND QUERY OPTIMIZATION

The global query is fragmented into sub-queries by a query decomposer. This function is normally performed by the Global Data Manager (GDM), which uses the distributed (or centralized, for some of the prototypes) data dictionary as a guide. The query decomposition strategy of heterogeneous systems is similar to that of homogeneous systems [GL184] except that, a language-to-language translation is required to mitigate the problem of data model differences [DEE87b]. The query processing procedure for

MULTIBASE systems is shown in Figure 6. The overall query processing responsibility is shared by the Global Data Manager (GDM) and the Local Data Interface (LDI). An LDI sub-system is located at each site. The GDM on receiving a DAPLEX global query references the DAPLEX global schema. It produces as output a DAPLEX global query that references DAPLEX local schema and the auxiliary database schema. In the next decomposition step, the GDM takes a transformed DAPLEX global query and separates it by local database sites, thus producing DAPLEX single site-queries. Other steps performed by the GDM include query optimization, and supervision of distributed query execution.

Many local host systems do not support all the capabilities provided by DAPLEX. For example, many systems do not support arithmetic expressions and compare operations. GDM ensures that each single-site query sent to an LDI can be handled in its entirety by the local host systems. The LDIs at the local sites are designed to be simple processors and not general purpose DBMSs. However, the capabilities of the LDIs may vary. If a local host DBMS provides restricted query capabilities, the system designers may choose to install a powerful LDI consisting of network interface, optimizer, translator and data formatter(see Figure 7).

A user or a control process can express a transaction in the Global Data Manipulation Language (GDML). The user process

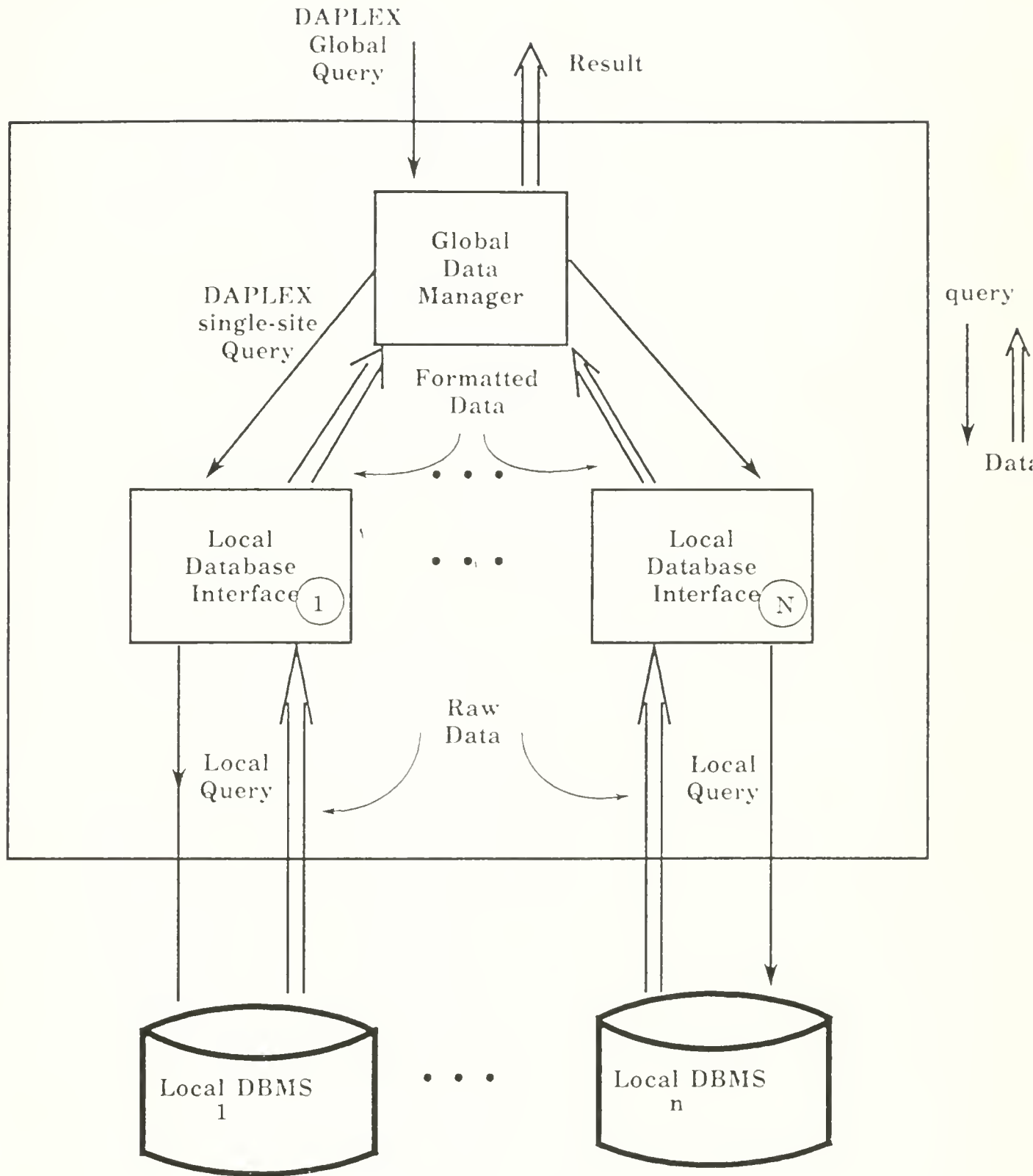


Figure 6: MULTIBASE query processing.

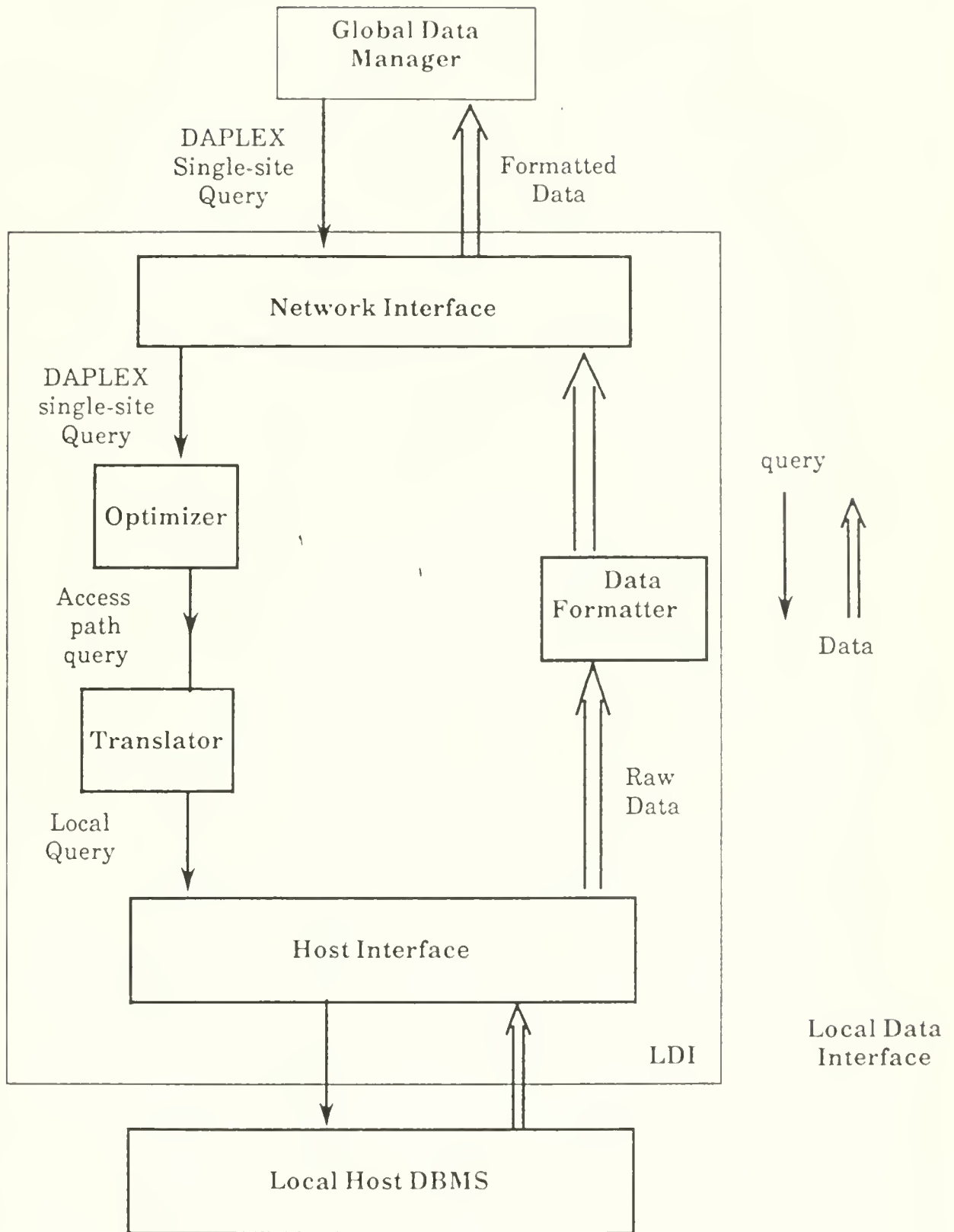


Figure 7: MULTIBASE Local Database Interface Architecture.

initiates a GDML query to the global external view of the integrated database. To process this query, IMDAS modifies the query tree so that the query operation operates on the data defined in the global conceptual view.

To process queries, query statements written in the GDML are embedded in COBOL and precompiled. After precompilation, source code files are sent to their respective local hosts for compilation.

A query in ADDS is addressed to a CDB. Further, it is translated into subqueries addressed to local LDBs. Each subquery is translated from the ADDS query language into the query language required by the specific DBMS. The ADDS directory contains the information required by the query optimization process.

The query optimization involves close examination of different processing methodologies at various levels. For example, GDM in the case of MULTIBASE, determines an overall strategy before forwarding the query. The final output is a global execution strategy that combines local processing and data movement steps. At the level of LDIs, the optimizer examines the DAPLEX single-site query and determines a local query processing strategy that will rapidly answer the query. The various DDBMS prototypes serve as useful models for processing queries and query optimization [BER81], [DAN82], [BOR81]. Also, [APE83], [YU

83], [DAY83], and [HWA82] discuss query optimization in a distributed heterogeneous environment.

IV.1.3 DISTRIBUTED TRANSACTION MANAGEMENT

Distributed Transaction Management (DTM) involves controlling the execution of distributed transactions while preserving the consistency of the common shared data, in spite of multiple transactions accessing the same set of data.

Most prototypes presently provide a retrieve-only interface to the DHDBMS. PRECI* and MERMAID allow global updates to be submitted to local DBMSs. The Transaction Manager (TM) component within IMDAS, performs the control and management of distributed transaction processing, including enforcement of database integrity, concurrency control and recovery. The integrity control mechanism of the TM realizes a serial execution schedule for each transaction. Transaction programs within NDMS are implemented as application programs of the host DBMS. The NDMS transaction processor provides facilities to invoke transaction programs, to support user interface, to exchange messages between application programs, as well as to synchronize transaction commit operations. In the case of IISS, all information within the system, is exchanged through messages. The Network Transaction Manager performs the task of receiving and interpreting messages. It also monitors and records the system

status. For most prototype systems, in the case of replicated data, update is performed only on the site of the original copy and then these sites broadcasts it to other copies.

The possibility of providing an additional layer of a software system to facilitate concurrency control and recovery, without disturbing the existing heterogeneous DBMSs, has shown some promise [MAD87]. In this approach, transactions are classified as those which issue updates for other sites, and those which execute at the sites where updates are required to be made. The DTM service interacts with local DBMSs to implement the desired update activity. The design incorporates queued transaction processing based on message oriented transaction processing protocols [GRAY86]. Further work in the area of transaction processing is in progress in the case of ADDS, IMDAS, MERMAID, IISS and at the level of various research groups for dealing with concurrency control and recovery issues for the DHDBMSs [GLI86].

IV.1.4 NETWORK SERVICES

The participating sites within an heterogeneous environment are connected by means of a communication network. The network is managed by system software called network operating system (NOS), which consists of local host operating systems and additional software incorporated at each host. The added software often

includes modifications to the local host operating systems. The participating hosts act independently and various degrees of sharing are possible through basic facilities such as remote procedure call, message passing, naming and access control. In addition there are services like filing, mail transfer, and remote computation. Similar to network services provided by NOS, Distributed Operating Systems (DOS) try to manage the resources of the network in a global fashion. These systems support operating system functions such as communication, access control, naming, and data storage and retrieval for the interacting application programs. This concept is used in IISS which offers an Operating System component called NTM (Network Transaction Manager), which supports application programs [IIS83] [TEM87].

In the case of MERMAID system, the controller component does process initiation of the drivers at local or remote sites, sets up interprocess communication, and handles the asynchronous input/output between the distributor and the drivers. The controller and the communication mechanism provide the functionality of a DOS above the independent UNIX 4.2 OS resident at the participating hosts. In the case of MULTIBASE both the GDM component and the local LDIs have a Network Interface sub-system, that establishes and maintains communications.

The IMDAS architecture presumes that factory data network utilizes a bus or ring local area network topology. The four

lowest layers of the ISO/OSI reference model must be implemented for each component system [ISO81]. The network communication function is performed by a Network Interface Process (NIP) in each component. The NIP maps the needed area of a conceptual global shared memory into the local shared memory. The mapping is done by replicating the contents of data into the shared memory areas of the remote components which require the copies of the data. PRECI* manages mail files at each site for receiving and sending messages. For most systems that build a layer of services on top of the existing network services, please refer to [TAN85], and [ROT85] for more details.

IV.1.5 DATA SECURITY ISSUES

In order to utilize the DHDBMS, a user request must be examined for both user authentication and access authorization. First, a user needs to be identified as an authentic user. Second, an authenticated user may not have access to the entire set of information resources available within the DHDBMS. Individual user access requests have to be processed by authorization routines that verify whether such an access is permissible to a user or not.

The database systems participating as components in a DHDBMS lack the ability to identify authentic users from other hosts. User authentication can be performed in two ways. In the

centralized approach the authentication information and procedures are kept in one place. It eases modification and protection of this information, but there is a dependence on the availability of the central system component with authentication function. In a decentralized approach the authenticated information is partially or fully replicated. It leads to problems of consistency and larger number of entries for misuse.

In conventional authorization schemes [GRI76], the authorization information is kept in tables and a set of rules is needed to assign access rights between users and objects. The two approaches traditionally used for authentication, namely access control lists and capabilities, presume a conceptual matrix in which a row corresponds to a user and a column corresponds to an object known to the system, such as a disk file, a relation, or a stored query. The intersection of the row and the column in this matrix indicates the particular user's accessibility to the specific object. In the context of a DHDBMS, an extension of this concept implies an exhaustive matrix of all users and all resources. Alternative approaches need to be explored for future systems.

V. RECOMMENDATIONS AND CONCLUSIONS

The intent of a DHDBMS is to provide a logically integrated user interface to physically non-integrated, distributed,

heterogeneous databases. This process of integration encompasses command and data translation, retrieval of information, and transaction management for multisite updates.

In order to present a single logical database to the user, a global schema is created. Operations on the global schema are translated into corresponding operations on local DBMSs. Creation of a global schema is difficult even when the number of participating databases is small because of several problems [LIT86]. First, the architectures of the local databases vary a great deal from each other [MAN83]. Second, semantic conflicts usually exist between local DBMSs. If the local databases disagree about a value, there may not be a single integrated value satisfactory to all users. Finally, a single global schema may not be possible when the number of participating databases is large.

In the area of transaction management, a DHDBMS involves incorporation of a concurrency control mechanism and a recovery mechanism, neither of which should interfere with existing mechanisms for local databases. Since one of the objectives of a DHDBMS is to provide complete autonomy to local DBMS sites, any change to existing mechanisms for concurrency control and recovery at local DBMSs must be ruled out. A complete solution to the maintenance of global consistency while permitting global-updates, is an area that requires sustained

effort [GLI84]. Further, adaptive control techniques must be employed to deal with failures, to support time-critical transactions, and to provide support for replication of information.

Most prototype DHDBMS rely on local host operating system capabilities to provide the required services through an additional layer of software. Based on the requirements, Tanenbaum [TAN82] emphasizes the following desirable features of distributed operating systems that aim at providing network support services:

- (a) The supporting systems should provide transaction oriented services rather than connection oriented services;
- (b) The system should enable the DBMS to exercise adequate control over input and output buffering systems; and
- (c) The system should permit reading and writing of isolated disk blocks with no implied relationships between successive disk transfers.

In our opinion, specific areas requiring further research work are as follows :

- (a) Development of automatic mapping tools for providing command and data translation to cater to various data models, languages, query structures, and data structures;
- (b) Design of semantic mapping aids such as semantic

- specification tools for languages [ULL87], [MOR68], [BRO84] and models for representing data semantics [HAM78], [SMI77], [BRO84];
- (c) Identification of strategies for query processing and query optimization based on knowledge processing techniques [HWA82], [YU 83], [DAY83];
 - (d) Developments of methods for providing and resolving access authorization in a heterogeneous environment; and
 - (e) Identification of better strategies for supporting initiation, migration and termination of distributed transactions [ROT85].

Advances in broad research areas such as Knowledge-based Engineering, Database Systems, Computer Graphics, and Distributed Operating Systems will continue to influence and catalyze the growth of Distributed Heterogeneous systems. Through sharing of ideas and concepts from related disciplines, the next generation of distributed heterogeneous database systems will offer a significantly higher level of performance, reliability, and flexibility than that available today.

REFERENCES

- [APE83] Apers, P.M.G., A.R. Henver, and S.B. Yao, "Optimization Algorithms for Distributed Queries", IEEE Transactions on Software Engineering, Jan. 83, pp. 57-68.
- [APP85] Appleton, D. S., "The Technology of Data Integration", Datamation, November 1, 1985, pp. 106-116.
- [BAR86] Barkmeyer, Edward, Mary Mitchell, K. P. Mikkilineni, Stanley Y. W. Su, and H. Lam, "An Architecture for Distributed Data Management in Computer Integrated Manufacturing", NBSIR 86-3312, NBS, January 1986.
- [BER81] Bernstein, P.A., N. Goodman, E. Wong, C.L. Reeve, and J.B. Rothnie, "Query Processing in a System for Distributed Databases (SDD-1)", ACM Trans. on Database Systems, Dec. 81, pp. 602-625.
- [BOR81] Borr, A., "Transaction Monitoring in ENCOMPASS : Reliable Distributed Transaction Processing", Proc. of the 7th Intl. Conf. on Very Large Data Bases, 1981, pp. 155-165.
- [BRE84] Breitbart, Y. J. and L. R. Tieman, "ADDS -Heterogeneous Distributed Database System", 3rd Int. Seminar on Distributed Data Sharing Systems, Italy, March 1984, Elsevier, North-Holland, 1985., pp. 7-24.
- [BRE86] Breitbart, Y. J., P. L. Olson, and G. R. Thompson, "Database Integration in a Distributed Heterogeneous Database System", IEEE Int. Conf. on Data Engineering, Los Angeles, February, 1986, pp. 301-310.
- [BRO86] Brodie M., "Database Management : A Survey", in Knowledge Base Management Systems, M. L. Brodie and J. Mylopoulos (Eds), Springer-Verlag, 1986.
- [CHE76] Chen, P.P.S., "The Entity-Relationship Model : Towards a Unified View of Data", ACM Transactions on Database Systems, Vol. 1, No. 1, March 1976, pp. 9-36.
- [DAN82] Daniels, D.P., P. Selinger, L. Haas, B. Lindsay, C. Mohan, A. Walker, and P. Wilms, "An Introduction to

Distributed Query Compilation in R*", Distributed
Databases, H.J. Schneider (Ed.), Sept. 82, pp. 291-309.

- [DAY86] Dayal, U. and J. N. Smith, "PROBE: A Knowledge-Oriented Database Management System", in Knowledge Base Management Systems, M. L. Brodie and J. Mylopoulos (Eds), Springer-Verlag, 1986.
- [DAY83] Dayal, U., "Processing Queries over Generalization Hierarchies in a Multidatabase System", Proc. of 9th Int. Conf. on Very Large Databases, Italy, 1983.
- [DEE81] Deen, S. M., et al, "The Design of a Canonical Database (PRECI)", The Computer Journal, Vol. 24, No. 3, 1981.
- [DEE84] Deen, S. M., R. R. Amin, and M. C. Taylor, "Query Decomposition in PRECI*," 3rd Int. Seminar on Distributed Data Sharing Systems, Italy, March 1984, Elsevier, North-Holland, 1985., pp.92-103
- [DEE85] Deen, S. M., R. R. Amin, G. O. Ofori-Dwumfuo, and M. C. Taylor, "The Architecture of a Generalised Distributed Database System - PRECI*", Computer Journal, Vol. 28, No. 3, 1985, pp. 282-290.
- [DEE87a] Deen, S.M., R. R. Amin, M. C. Taylor, "Implementation of a Prototype for PRECI*", Computer Journal, Vol. 30, No. 2, 1987.
- [DEE87b] Deen, S.M., R. R. Amin, M. C. Taylor, "Data Integration in Distributed Databases", IEEE Transactions on Software Engineering, July 1987.
- [DEV82] Devor, C., R. Elmasri, J.A. Larson, S. Rahimi and J.P. Richardson, "Five Schema Architecture Extends DBMS to Distributed Applications", Electronic Design, March 18, 1982, pp. ss27-ss32.
- [GLI84] Gligor, V. D. and G. L. Lukenbaugh, "Interconnecting Hetrogeneous Database Management Systems", Computer, January 1984, pp. 33-43.
- [GLI86] Gligor, V. D. and R. Popescu-Zeletin, "Transaction Management in Distributed Hetrogeneous Database Management Systems", Information Systems, Vol. 11. No. 4, pp. 287-297, 1986.
- [GOL84] Goldhisch, D., T. landers, R. L. Rosenberg and L. Yedwab,

"MULTIBASE System Administrator's Guide", Tech. Rep.,
CCA, Cambridge, MA, November 1984.

- [GRAY86] GRAY, J. "An Approach to Decentralized Computer Systems", IEEE Transactions on Software Engineering, Vol. SE-12, No. 6, June 86, pp. 684-692.
- [GRI76] Griffiths, P. P. and B. W. Wade, "An Authorization Mechanism for a Relational Data Base System", Transactions on Database Systems, Vol. 1, No. 3, pp. 242-255, Sept. 1976.
- [HAM78] Hammer, A., and D. McLeod, "The Semantic Data Model", Proc. of the SIGMOD Conf., 1978.
- [HWA82] Hwang, H.Y., "Database Integration and Query Optimization in Multi-database systems, Ph.d. thesis, Dept. of Computer Science, University of Texas at Austin, Austin, Texas, U.S.A..
- [IIS83] Integrated Information Support System (IISS) report, "Integrated Computer-Aided Manufacturing (ICAM)", Materials Laboratory, Air Force Systems Command, Wright-Patterson AFB, February 1983.
- [ISO81] "Data Processing - Open Systems Interconnection - Basic Reference Model", ISO standard 7498, International Standards Organization, Geneva, 1981.
- [KRI85] Krishnamurthy, P., "A Data Manipulation Language for the Semantic Association Model, SAM*", Masters Thesis, College of Engineering, University of Florida, Gainesville, FL, 1985.
- [LAN82] Landers, T. and R. L. Rosenberg, "An Overview of MULTIBASE", Second Symp. Distributed Database, Berlin, Sept. 1982, North-Holland, New York, 1982, pp. 311-366.
- [LIB86] Libes, Don and Ed Barkmeyer, "IMDAS - An Overview", Draft Report, Integrated Systems Group, NBS, Gaithersburg, MD, 1986.
- [LIT85] Litwin, W., "An Overview of the Multidatabase System MRDSM", ACM National Conf., Denver, October 1985.
- [LIT86] Litwin, W. and Abdelaziz Abdellatif, "Multidatabase Interoperability", Computer, December 1986, pp. 10-18.

- [MAC85] MacGregor, R., "ARIEL - A Semantic Front-End to Relational DBMSs", in Proc. of VLDB, August 1985.
- [MAD87] Madnick, S.E., et al, "Concurrency Control and Recovery in Distributed, Hetrogeneous Database Management Systems., working paper, 1987.
- [MAN83] Manola, F.A., "Model to Model Mappings and Conversion in a Family of Data Model Specifications", Computer Corporation of America report No. CCA-83-14, Dec. 83.
- [MOR68] Morris, J.H., "Lamda Calculus Models of Programming Languages", Doctoral Thesis , M.I.T., Cambridge, MA02139, U.S.A., 1968.
- [ROT85] Rothermel, K., "Communication Primitives Supporting the Execution of Distributed Transactions", Proc. of the 18th Annual Hawaii Intl. Conf. on System Science", 1985, pp. 274-284.
- [SHI81] Shipman, D., "The Functional Data Model and the Data Language DAPLEX", ACM Trans. on Database Systems, March 1981.
- [SMI77] Smith, J.M., and D.C.P. Smith, "Database Abstractions : Aggregations and Generalization", IEEE Transactions on Database Systems, Vol. 2, No. 2, June 1977.
- [SMI81] Smith, J. M. et. al., "Multibase - Integrating Heterogeneous Distributed Database Systems", Proc. of AFIPS, Vol. 50, 1981, pp. 487-499.
- [STA84] Staniszkis, W. Kaminski, M. Kowalewski, K. Krajewski, S. Mezyk, and G. Turco, "Architecture of the Network Data Management System," 3rd Int. Seminar on Distributed Data Sharing Systems, Italy, March 1984, Elsevier, North-Holland, 1985., pp. 57-74.
- [SU 83] Su, Stanley, "SAM* : A Semantic Association Model for Corporate and Scientific-Statistical Databases", Information Sciences, Vol. 29, 1983, pp. 151-199.
- [TAN85] Tanenbaum, A.S., and R. V. Renesse, "Distributed Operating Systems", Computing Surveys, vol. 17, No. 4, Dec. 85.
- [TAN82] Tanenbaum, A.S., and S.J., Mullender, "Operating System Requirements for Distributed Data Base Systems," In

Distributed Databases, H.-J. Schneider, Ed. North-Holland
Publ., Amsterdam, pp. 105-114.

- [TEM86] Templeton, M., D. Brill, A. Chen, S. Dao, and E. Lund,
"MERMAID - Experiences With Network Operation," IEEE
Int. Conf. on Data Engineering, Los Angeles, February
1986, pp. 292-300.
- [TEM87] Templeton, M. et al, "MERMAID - A Front-end to
Distributed Heterogeneous Databases", invited paper,
Proceedings of the IEEE, may 1987, pp. 695-708.
- [ULL87] Ullman, J.D., A.V. Aho, and R. Sethi, "Principles of
Compiler Design, Addison-Wesley, Reading, Mass. 1987.
- [WON84] Wong, K. K. and P. Bazex, "MRDSM: A Relational
Multidatabases Management System", 3rd Int. Seminar on
Distributed Data Sharing Systems, Italy, March 1984,
Elsevier, North-Holland, 1985, pp 77-85.
- [YU 83] Yu, C.T. and C.C. Chang, "On the Design of a Query
Processing Strategy in a Distributed Database
Environment", Proc. of ACM SIGMOD, 1983, pp. 30-39.
- [YU 85] Yu, C, C. Chang, M. Templeton, D. Brill, and E. Lund,
"Query Processing in a Fragmented Relational Distributed
System: MERMAID," IEEE Trans. on Software Engineering,
August 1985.

BIBLIOGRAPHY

- [APP86a] Appleton, D. S., "Very Large Projects", Datamation, January 15, 1986, pp. 63-70.
- [APP86b] Appleton, D. S., "Information Asset Management", Datamation, February 1, 1986, pp. 72-76.
- [APP86c] Appleton, D. S., "Rule-Based Data Resource Management", Datamation, May 1, 1986, pp. 86-99.
- [BAT86] Batni, C., M. Lenzerini, and S.B., Navathe, "A Comparative Analysis of Methodologies for Database Schema Integration", ACM Computing Surveys, Vol. 18, No. 4, Dec. 86.
- [BRE84] Breitbart, Y. J., L. F. Kemp, G. R. Thompson, A. Silberschatz, "Performance Evaluation of a Simulation Model for Data Retrieval in a Heterogeneous Database Environment", Proc. of IEEE, Trends & Applications 1984, pp. 190-197.
- [BRE85] Breitbart, Y. and P. Paolini, "The Multibase Session", in Distributed Data Sharing Systems, F. A. Schreiber and W. Litwin (Eds.) Elsevier (North-Holland), 1985, pp. 3-6.
- [CAR85] Cardenas, A.F., and Wang, G.R., "Translation of SQL/DS Data Access/update into Entity-relationship Data Access/update", Proc. of the 4th IEEE Intl. Conf. on Entity-Relationship Approach, Oct. 85.
- [CAR87] Cardenas, A.F., "Hetrogeneous Distributed Data Base Management The HD-DBMS", invited paper, Proceedings of the IEEE, may 1987, pp. 588-600.
- [CHI84] Chimia, J. L., "Query Decomposition in a Distributed Database System Using Satellite Communications", 3rd Int. Conf. on Distributed Data Sharing Systems, Italy, March 1984, Elsevier (North-Holland), pp. 105-118.
- [CHU86] Chung, L. V., M. D. Yanike, E. J. Johnson and E. J. Byrne, "Design and Implementation of a Relational Database Server In a Hetrogeneous Network Environment", IEEE Int. Conf. on Data Engineering, Los Angels, California, FEB. 5-7, 1986, pp. 685-692.

- [DUT86] Dutton, D.L., "In Pursuit of CIM, Datamation, February 1986, pp. 63-66.
- [ELM86] Elmasri, R., Larson, J. and Navathe, S., "Schema Integration Algorithms for Federated Databases and Logical Database Design", Dept. of Computer Sc., University of Houston, Houston, TX 77004.
- [FER82] Ferrier, A., C. Stangret, "Heterogeneity in the Distributed Database Management System SIRIUS-DELTA", Proceedings of 8th VLDB, Mexico City, Sept. 1982.
- [GLI85] Gligor, V. D. and R. Popescu-Zeletin, "Concurrency Control Issues in Distributed Heterogeneous Database Management Systems", 3rd Int. Seminar on Distributed Data Sharing Systems, Italy, March 1984, Elsevier, North-Holland, 1985, pp 43-56.
- [HEI85] Heimbigner, D., and McLeod, D., "A Federated Architecture for Information Management", ACM Trans. on Office Information Systems, Vol. 3, No. 3, July 85, pp. 253-278.
- [KEL86] Keller, A.M., "The Role of Semantics in Translating View Updates", Computer, IEEE magazine, Jan. 86.
- [KIM81] Kimbleton, S. R. and P. Wang, "Applications and Protocols", Distributed Systems: Architecture and Implementation, Lecture Notes in Computer Science, Paul, Lampson and Siegert, (eds.) Vol. 105, Springer Verlag, New York, 1981, pp. 308-370.
- [LEF84] Lefons, E., and A. Silvestri, "The Use of Multidatabase in Decision Support Systems", Proc. of the 3rd Intl. Seminar on Distributed Data Sharing Systems, Italy, March 84, Elsevier, North-Holland, 1985.
- [LIT82] Litwin, W. et al. "SIRIUS Systems for Distributed Data Management", in Distributed Data Bases, H. J. Schneider (ed), North-Holland, 1982, pp. 311-366.
- [LIT87] Litwin, W. and A. Abdellatif, "An Overview of the Multidatabase Manipulation Language MDSL, invited paper, Proceedings of the IEEE, May 1987, pp. 621-632.
- [LOG81] Leveson, N. G. and A. I. Wasserman, "Logical Decentralization and Semantic Integrity in a Distributed Information System", 2nd. Intl. Seminar on Distributed Data Sharing Systems, Netherlands, June 1981, Elsevier

(North-Holland), 1982, pp. 243-253.

- [MAL84] Malkanoff, M.A., "The CIMS Database : Goals, Problems, Case Studies and Proposed Approach Outlined", Industrial Engineering, November 1984, pp. 78-93.
- [NAV84] Navathe, S.B., Shashidhar, T., and Elmasri, R., "Relationship Merging in Schema Integration", Proc. of 10th Intl. Conf. on Very Large Data Bases, Aug. 84, pp. 78-90.
- [NAV86] Navathe, S., Elmasri, R., and Larson, J., "Integrating User Views in Database Design", Computer, IEEE Magazine, Jan. 86.
- [PRA86] Pratt, S. J., "The Alchemy Model: A Model for Homogeneous and Hetrogeneous distributed Computing System", Operating Systems Review, Vol. 20, No. 2, April 1986, pp. 25-37.
- [PU 87] Pu, Calton, "Superdatabases for Composition of Heterogeneous Databases", Department of Computer Science, Columbia University, Technical Report No. CUCS-243-86, June 1987.
- [SPA81] Spaccapietra, S., B. Demo, A. Di Leva, C. Parent, C. Perez De Celis and K. Belfar, "An Approach to Effective Hetrogeneous Databases Cooperation", 2nd Int. Seminar on Distributed Data Sharing Systems, Netherlands, June 1981, Elsevier (North-Holland), 1982, pp. 209-218.
- [STA86] Staniszkis, W. "Integrating Heterogeneous Databases", on Relational Databases in Infotech State of the Art Report, 1986.
- [TAK79] Takizawa, M., et al, "The Four-schema Concept as the Gross Architecture of Distributed Database and Heterogenity Problems", Journal of Information Processing (JIP) (IPSJ), 2 (1979), pp. 134-142.
- [TAK83] Takizawa, M., "Heterogeneous Distributed Database Systems : JDDBS", Database Engineering, 6, 1, (1983).
- [TEM83] Templeton, M., D. Brill, A. Hwang, I. Kameny and E. Lund, "An Overview of the Mermaid System - A Frontend to Heterogeneous Databases, Proceedings of EASCON 83, 1983, pp. 387-402.
- [TOJ85] Tojo, A. and T. Sato, "Interoperable Database System: A

New R&D Project and Its Impact on Multimedia Information Processing Technology", IEEE Workshop on Computer Architecture for Pattern Analysis and Image Database Management, Miami, FL, November, 1985, pp 336-339.

[WIE86a] Wiederhold, G., and XiaoLei Qian, "Modeling Asynchrony in Distributed Databases", invited paper, Computer Science Department, Stanford University, Stanford, CA 94305, USA., 1986.

Date Due	
NOV. 1 2 1992	
	Lib-26-67

MIT LIBRARIES



3 9080 005 239 774

